



Corso di Laurea Triennale in Informatica
Università Degli Studi della Basilicata

Sistemi Operativi

Docente:
ugo.erra@unibas.it

11° Lezione

Interfaccia del file system

Sommario della lezione



- Introduzione
- Concetto di file
- Operazioni sui file
- Metodi di accesso
- Struttura delle directory
- Protezione

Introduzione al file system

- Il **File system** rappresenta per gli utenti l'aspetto più visibile di un Sistemi Operativi
- Lo scopo di un file system è fornire un meccanismo per la memorizzazione, l'accesso ai dati ed alle applicazioni del Sistema Operativo verso utenti
- Un File System consiste di due parti:
 - ▣ Un insieme di **file**
 - ▣ Una **struttura di directory**, che permette di **organizzare** tutti i file del sistema

Il concetto di file - 1

- **Un file è unità logica di informazione memorizzata** permanentemente (di solito) su un supporto di memoria secondaria e dotato di:
 - ▣ Un nome
 - ▣ Una posizione logica all'interno del File System
 - ▣ Degli attributi (dimensioni, diritti di accesso, date di creazione, accesso e modifica, etc...)
- Un file può contenere diverse tipi di informazioni come:
 - ▣ Dati
 - ▣ Programmi
 - ▣ Documenti

Il concetto di file - 2



- Nonostante un file sia sempre una sequenza di bit il Sistema Operativo ne riconosce la struttura interna, infatti:
 - ▣ Un file di testo è formato da caratteri organizzati in righe
 - ▣ Un programma sorgente è suddiviso in procedure e dati
 - ▣ Un eseguibile è spesso suddiviso in segmenti

Attributi dei file

- Ad ogni file solitamente sono associati degli **attributi** la cui memorizzazione può richiedere anche alcuni kilobyte
- Gli attributi hanno lo scopo di facilitare la gestione dei file all'interno del sistema
- Alcuni tipici attributi sono:
 - ▣ **Nome simbolico** - unica informazioni umanamente leggibile
 - ▣ **Tipo** - necessaria per quei sistemi operativi che supportano diversi tipi di file
 - ▣ **Locazione** - un puntatore alla locazione del file sulla memoria secondaria (ossia, di solito, l'hard disk)
 - ▣ **Dimensione** - dimensione del file corrente
 - ▣ **Protezione** – informazioni di controllo per chi può leggere, scrivere ed eseguire il file
 - ▣ **Ora, data e identificazione dell'utente** – data di creazione del file e ultima modifica

Operazioni sui file - 1

- Un file può essere visto come un **tipo di dato astratto** definito solo dalle operazioni che si possono compiere su di esso e rese disponibili dal sistema operativo:
 - ▣ **Creazione di un file** – Il Sistema Operativo si incarica di trovare lo spazio per il file, e poi di creare un accesso al file attraverso la directory che “contiene” il file, secondo le modalità di accesso stabilite per quel file
 - ▣ **Scrittura di un file** – Il Sistema Operativo fornisce una chiamata di sistema per indicare il nome del file e le informazioni che si vogliono scrivere. Un puntatore è adoperato per indicare la posizione in cui avviene l’operazione di scrittura

Operazioni sui file - 2

- ▣ **Lettura di un file** - Anche in questo caso una chiamata di sistema fornisce un puntatore al file che indica il successivo blocco da leggere. Solitamente è possibile leggere e scrivere un file contemporaneamente ed il puntatore indica la posizione in cui leggere o scrivere
- ▣ **Riposizionamento di un file** – Questa operazione nota anche come seek permette di spostare il puntatore in una nuova posizione
- ▣ **Cancellazione di un file** – Il Sistema Operativo cerca il file e lo cancella liberando lo spazio associato al file
- ▣ **Troncamento di un file** – Cancella il contenuto di un file recuperando lo spazio ma mantenendo inalterati gli attributi
- ▣ **Rinominare, copiare, spostare**

Modalità di accesso

- L'accesso ad un file può essere fatto in due modalità:
 - ▣ **Accesso sequenziale** - I dati del file (nel caso più semplice, i byte di cui è composto) vengono letti o modificati in modo sequenziale, a partire dall'inizio del file
 - ▣ **Accesso diretto** - I dati possono essere letti o modificati in un punto ben preciso del file
 - Ad esempio, in un file di testo, vogliamo poter leggere la 1000-esima riga del testo
- L'accesso diretto può essere simulato attraverso quello sequenziale
 - ▣ Per leggere la 1000-esima riga del file possiamo incominciare a leggerlo dal primo carattere, contare le varie righe e fermarci quando abbiamo trovato la 1000-esima

Directory, cartelle o folder

- Poiché il numero dei file all'interno di un file system può crescere è necessario organizzare i file in modo da potervi accedere in tempi ragionevoli (oltre che in maniera intuitiva)
 - ▣ In particolare il tempo di accesso ai singoli file non deve crescere al crescere del numero dei file
- Le **directory** sono un modo per tenere traccia dei file all'interno di file system
- Una directory è essenzialmente un tabella di simboli che permette di risalire a tutte le informazioni relative ad un file (cioè i suoi dati e i suoi attributi)

Directory



- Le tipiche operazioni che un Sistema Operativo offre per le directory sono:
 - ▣ Ricerca di un file
 - ▣ Creazione/cancellazione di un file
 - ▣ Elenco del contenuto della directory
 - ▣ Cambiamento del nome di un file
 - ▣ Spostamento di un file in un'altra directory
 - ▣ Spostamento da una directory ad un'altra

Directory

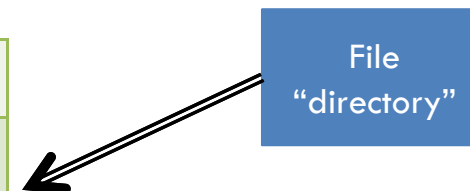


- Le informazioni contenute nella directory sono vitali per poter accedere ai file
 - ▣ La perdita dei dati della directory può compromettere l'accesso ai file
- Le directory devono essere logicamente organizzate in modo da fornire un minimo di efficienza nel recupero delle informazioni contenute

Directory

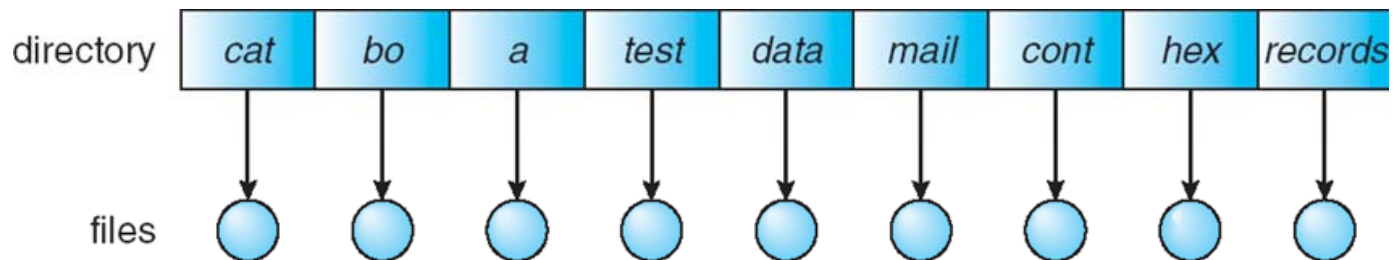
- In alcuni casi le directory sono dei veri e propri file che mantengono informazioni relative ai file in esso contenuti
- Un file “directory” è organizzato in una entry all’interno della quale troviamo informazioni quali il nome del file e gli attributi oppure un puntatore alla struttura che li contiene

word.exe	attributi
pipipo.txt	attributi
compleanno.jpg	attributi
maledetto_progetto.c	attributi



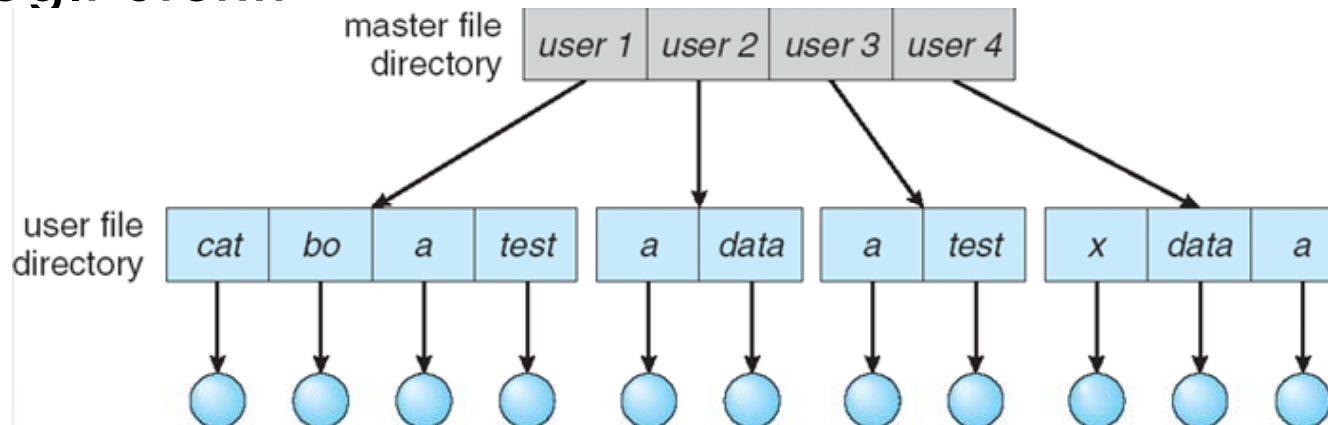
Directory ad un solo livello

- L'organizzazione dei file all'interno delle directory può essere gestita nel caso più semplice in un'unica directory che contiene tutti i file
 - ▣ File di utenti diversi non possono avere lo stesso nome
 - ▣ I file non possono essere raggruppati separatamente
 - ▣ La ricerca di un file può essere molto inefficiente



Directory a due livelli

- Un miglioramento consiste nell'avere una user file directory per ogni utente ed una master file directory che “punta” o “contiene” le directory degli utenti



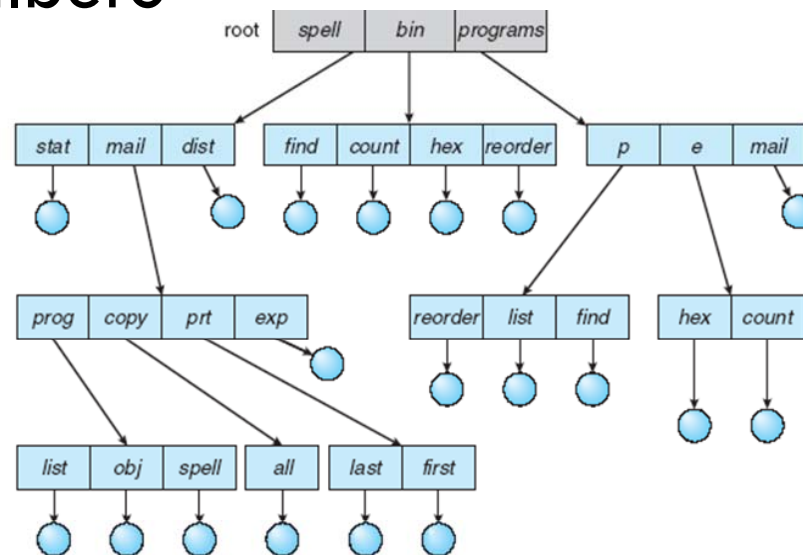
Pathname dei file



- Dalla directory a due livelli nasce il concetto di *pathname* per indicare il percorso del file a partire dalla master file directory
- Poiché i file sono organizzati separatamente anche la ricerca di un file all'interno de file system è più efficiente

Directory a più livelli - 1

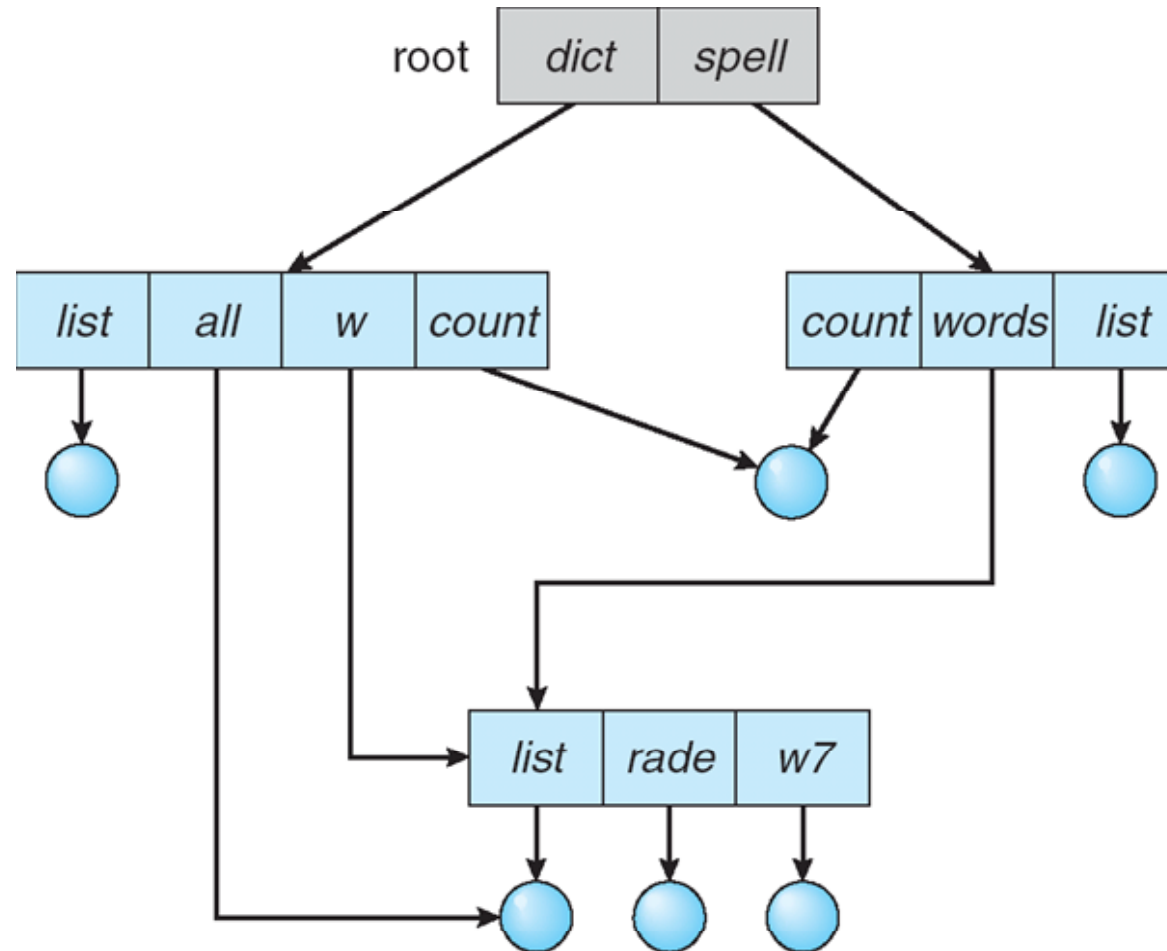
- La generalizzazione del concetto di directory a due livelli sono le directory a più livelli
- Una directory può contenere ricorsivamente altri file ed altre directory facendo assumere al file system una struttura ad albero



Directory a con struttura a grafo aciclico

- La struttura ad albero non permette di condividere file o directory con nomi diversi
 - ▣ Questo è un grosso limite alla condivisione e alla cooperazione
- Se lo si desidera, lo stesso file dovrebbe poter essere visto da directory diverse, possibilmente con nomi diversi nelle diverse directory
- I diversi collegamenti ad un file o ad una directory prendono il nome di **link**
- I Sistemi Operativi realizzano i link in maniera diversa, ottenendo risultati diversi

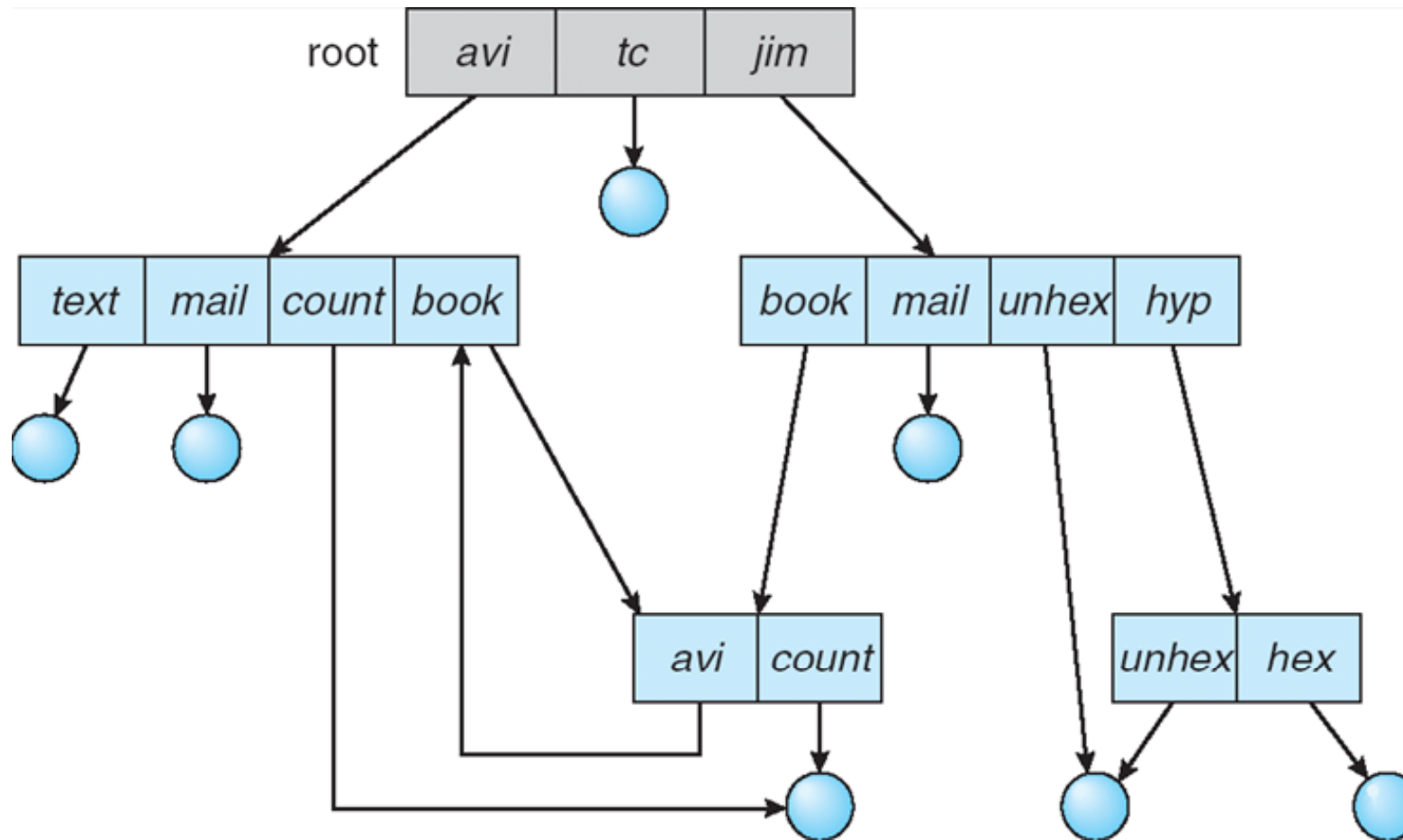
Directory a con struttura a grafo aciclico



Directory con struttura a grafo generale

- In una directory con struttura a grafo generale, una directory può contenere il nome di una directory padre
- Questa possibilità può causare problemi di loop quando si visita la directory e le sue sottodirectory ricorsivamente
- Una problema che potrebbe sorgere è quando cancellare il contenuto di una directory che contiene la directory padre

Directory con struttura a grafo generale



Accesso ai file

- Per accedere ad un file si utilizza normalmente il suo pathname
- Questo accesso risulta essere particolarmente inefficiente se dobbiamo accedere continuamente al file per le operazioni di lettura e scrittura
- Ad esempio, un programma che deve accedere al file in lettura

```
fscanf("c:/documenti/studenti/paolorossi", "%d", eta);
```

dovrà ogni volta utilizzare il pathname completo

Accesso ai file



- Per leggere il file “paolorossi” il Sistema Operativo ogni volta dovrebbe:
 - ▣ Accedere al disco fisso per prelevare le informazioni sulla directory “/” alla ricerca della cartella “documenti”
 - ▣ Dalla cartella documenti cercare la cartella “studenti”
 - ▣ Continuare in questo modo fino a raggiungere il file “paolorossi”

File table

- Per evitare che ad ogni accesso sia effettuata una operazione del genere il Sistema Operativo richiede di aprire (system call open) i file da utilizzare
- Per ogni file aperto viene restituito un descrittore che sarà copiato nel PCB del processo ed in particolare all'interno della **open file table**
- Ad ogni accesso al file il Sistema Operativo non passa più attraverso il file system
- Quando un programma chiude (system call close) un file su cui ha terminato di operare, le informazioni del file nella open file table possono essere rimosse

Protezione

- La protezione gestisce le informazioni contenute in un Sistema Operativo da danni fisici o accessi impropri
- In un sistema multiutente la protezione è una esigenza importante
- Il controllo offerto solitamente cerca di limitare i tipi di accesso possibili
 - ▣ Lettura
 - ▣ Scrittura
 - ▣ Esecuzione
 - ▣ Aggiunta
 - ▣ Cancellazione
 - ▣ Lista dei file in una directory

Controllo degli accessi

- In molti sistema l'accesso deve essere regolato a livello dell'identità dell'utente
- Una possibile soluzione consiste nell'**utilizzare una lista di controllo degli accessi** per ogni file o directory
 - ▣ Quando un utente richiede un accesso si controlla se l'utente è presente all'interno della lista di controllo degli accessi per quel file
- Questa soluzione ha diversi svantaggi:
 - ▣ La costruzione di una lista per ogni file può essere un compito tedioso
 - ▣ Per ogni file bisogna memorizzare delle informazioni aggiuntive che possono crescere

Controllo degli accessi basato su classi

- La lista di controllo degli accessi può essere implementata in maniera “ridotta”
- All’interno del sistema si individuano tre gruppi di utenti:
 - ▣ **Proprietario** - L’utente che ha creato il file
 - ▣ **Gruppo** - Un gruppo i cui membri condividono il file ed hanno accessi simili
 - ▣ **Universo** - Tutti gli altri utenti del sistema
- Ad esempio sotto linux il file

```
19 -rw-r--r--+ 1 alessandra studenti 130 May 25 22:13 esame
```

```
19 drwxr--r--+ 1 alessandra studenti 130 May 25 22:13 progetto
```

indica che l’utente `alessandra` appartenente al gruppo `studenti` possiede il file `esame` la directory `progetto`